IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of: Confirmation No.: 8375

Zhen Liu, et al. Examiner: Navneet K. Ahluwalia

Serial No.: 10/810,152 Group Art Unit No.: 2166

Filed: March 26, 2004

For: TECHNIQUES FOR MANAGING XML DATA ASSOCIATED WITH ULTIPLE EXECUTION UNITS

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed on August 7, 2009.

I. REAL PARTY IN INTEREST

Oracle International Corporation is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1-50 are pending in the application, and are the subject of this appeal. Claims 1-50 were finally rejected.

IV. STATUS OF AMENDMENTS

No amendments were filed after the Final Office Action.

V. SUMMARY OF CLAIMED SUBJECT MATTER

1. INDEPENDNET CLAIMS

The present application contains independent Claims 1, 39, 45 and 50. The independent claims are directed generally to techniques for managing XML data associated with multiple execution units.

Claim 1 recites (with added reference annotations in parenthesis) a method comprising the computer-implemented steps (page 9, paragraph [23], lines 1-5; FIG. 1) of:

detecting that a portion of a query execution plan to service a request for data will cause a first producer execution unit that will perform said portion (page 10, paragraph [27], lines 1-6; FIG. 1, element 102), according to said query execution plan (page 10, paragraph [27], lines 1-6; FIG. 1, element 102), to generate XML data for use by a second consumer execution unit in performing another portion of said query execution plan (page 10, paragraph [27], lines 1-6; FIG. 1, element 102);

generating information to send to said first execution unit to cause said first execution unit to perform said portion of said query execution plan (page 10, paragraph [28], lines 1-6; FIG. 1, element 104);

wherein said information would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit (page 11, paragraph [31], lines 1-8); and

annotating said information with an annotation that causes XML data generated by said first execution unit to be transformed to a canonical form for use by said second execution unit in performing said another portion of said query execution plan (page 10, paragraph [29], lines 1-6; FIG. 1, element 106),

wherein said annotating causes removal of one or more references to execution unit-specific data that is accessible by the first execution unit but that is not accessible by the second execution unit (page 8, paragraph [20], lines 8-13).

Claim 39 recites (with added reference annotations in parenthesis) A method for processing XML data (page 9, paragraph [23], lines 1-5; FIG. 1), comprising the computer-implemented steps of:

receiving information at a first execution unit to cause said first execution unit to perform work associated with a query execution plan for servicing a request for data (page 10, paragraph [27], lines 4-6; FIG. 1, element 102);

wherein said information comprises an annotation that causes the XML data generated by said first execution unit to be transformed to a canonical form for use by a second execution unit (page 10, paragraph [29], lines 1-6; FIG. 1, element 106);

wherein said information, without said annotation, would cause said second execution unit to receive from said first execution unit XML data in a first form that cannot be used by said second execution unit (page 11, paragraph [31], lines 1-8);

transforming XML data generated by said first execution unit to said canonical form prior to providing said XML data to said second execution unit (page 4, paragraph [12], lines 1-8),

wherein transforming XML data comprises removing one or more references to execution unit-specific data that is accessible to the first execution unit but that is not accessible to the second execution unit (page 8, paragraph [20], lines 8-13); and

providing XML data that is transformed to said second execution unit in said canonical form for use in performing work associated with said query execution plan by said second execution unit (page 10, paragraph [29], lines 1-4).

Claim 45 recites (with added reference annotations in parenthesis) A database system (page 7, paragraph [18], lines 1-5; page 5, paragraph [13], lines 6-9; page 5, paragraph [15], lines 1-5) comprising:

a query optimizer (page 8, paragraph [21], lines 1-6) that receives a database query, formulates a query plan based on said query, and sends information based on said plan to a first execution unit (page 8, paragraph [21], lines 1-6);

wherein formulating a plan includes determining that said first execution unit produces XML data for use by a second execution unit (page 9, paragraph [26], lines 1-6), and determining whether said first execution unit produces said XML data in a first form that said second execution unit is able to use (page 10, paragraph [27], lines 1-6; FIG. 1, element 102);

said first execution unit that receives said information from said query optimizer (page 10, paragraph [28], lines 1-6; FIG. 1, element 104); and

said second execution unit that receives said XML data from said first execution unit (page 10, paragraph [29], lines 1-6; FIG. 1, element 106).

Claim 50 recites (with added reference annotations in parenthesis) A system (page 13, paragraph [37], lines 1-6; FIG. 2) comprising:

means for detecting that a portion of a query execution plan to service a request for data will cause a first producer execution unit that will perform said portion, according to said query execution plan, to generate XML data for use by a second consumer execution unit in performing another portion of said query execution plan (page 10, paragraph [27], lines 1-6; FIG. 1, element 102);

means for generating information to send to said first execution unit to cause said first execution unit to perform said portion of said query execution plan (page 10, paragraph [28], lines 1-6; FIG. 1, element 104);

wherein said information would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit (page 11, paragraph [31], lines 1-8); and

means for annotating said information with an annotation that causes XML data generated by said first execution unit to be transformed to a canonical form for use by said

second execution unit in performing said another portion of said query execution plan (page 10, paragraph [29], lines 1-6; FIG. 1, element 106),

wherein said annotating causes removal of one or more references to execution unit-specific data that is accessible by the first execution unit but that is not accessible by the second execution unit (page 8, paragraph [20], lines 8-13).

2. DEPENDENT CLAIM 2

The applicants also argue dependent Claim 2. Claim 2 further limits the step of generating information, recited in Claim 1, and is directed generally to generating information that, prior to annotation, would cause the first execution unit to generate the XML data in a form that cannot be used by the second execution unit.

Claim 2 recites (with added reference annotations in parenthesis) The method of Claim 1,

wherein the step of generating information includes generating information that, prior to annotating said information, would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit (page 11, paragraph [31], lines 1-8), and

wherein said canonical form is different from said first form (page 11, paragraph [31], lines 1-8).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. The rejections of Claims 1-50 under 35 U.S.C. § 102(e) as allegedly anticipated by Fernandez et al. (U.S. Patent 6, 785,673) ("*Fernandez*"). (Note: the Office

Actions received in this case contradict each other as to whether Fernandez et al. does or does not anticipate Claim 1. However, the most recent Office Action contains the rejection).

VII. ARGUMENT

A. INTRODUCTION

"A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). To show anticipation under 35 U.S.C. § 102, "[it] is not enough that the prior art reference discloses part of the claimed invention, which an ordinary artisan might supplement to make the whole, or that it includes multiple, distinct teaching that the artisan might somehow combine to achieve the claimed invention." *In re Arkley*, 455 F.2d 586, 587 (CCPA 1972) "[T]he prior art reference must clearly and unequivocally disclose the claimed [invention] or direct those skilled in the art to the [invention] without any need for ***picking, choosing, and combining*** various disclosures not directly related to each other by the teaching of the cited reference." *Id.* In *Net Moneyin, Inc. v. Verisign, Inc.*, No. 2007-1565, U.S. CAFC, 2008, the Federal Circuit specifically held that unless a reference discloses within the four corners of the document not only all of the limitations claimed but also all of the limitations arranged or combined in the same way as recited in the claim, it **cannot** be said to prove prior invention of the thing claimed and, thus, cannot anticipate under 35 U.S.C. § 102. (Emphasis added)

In the present matter, the Examiner has made clearly erroneous factual findings regarding the scope and content of the prior art, and in particular, what the *Fernandez'* reference teaches. Therefore, the Examiner's analysis, and the rejection based thereon, are invalid.

With respect to the present application, it is respectfully submitted that *Fernandez* does not anticipate all the limitations of independent Claims 1, 39 45 and 50. It is further

submitted that the Examiner has not proffered a sufficient factual basis during the prosecution of the present application to support the rejection of Claims 1, 39 45 and 50 under U.S.C. § 102(e) as being anticipated by *Fernandez*.

B. THE OFFICE ACTIONS CONRADICT EACH OTHER

The Office Actions are full of contradictions with respect to what they allege is shown in *Fernandez* et al. (U.S. Patent 6,785,673). Specifically, the present Final Office Action, mailed on June 8, 2009, alleges that *Fernandez* anticipates all limitations received in Claim 1. However, the Examiner explicitly (and correctly) acknowledged in two previous Office Actions that *Fernandez* fails to anticipate Claim 1.

For example, in the Office Action mailed on November 1, 2007, the Office acknowledged that *Fernandez* does not teach the producer unit and the consumer unit, both of which are recited in independent Claim 1. The Office Action could not have been clearer on the issue: "**Fernandez does not teach the producer unit and consumer unit as disclosed**" in Claims 1 and 20.

The same has been admitted in the Office Action mailed on May 29, 2008, which repeats the prior admission that: "**Fernandez does not teach the producer unit and consumer unit as disclosed**". The Office's correct understanding that *Fernandez* is missing key limitations of Claim 1, which has been reaffirmed in two previous Office Actions, contradicts the present §102(e) rejection of Claim 1.

This error has resulted in faulty rejections.

C. THE REJECTIONS OF CLAIMS 1-50 UNDER 35 USC 102(E) AS BEING ANTICIPATED BY FERNANDEZ ARE BASED ON A CLEAR FACTUAL ERROR

The rejection of Claims 1-50 are based on a clear factual error. The Final Action,

mailed on June 8, 2009, alleges that *Fernandez* anticipates the limitations of Claims 1-50. This is incorrect. (Even though only Claim 1 is discussed below, the arguments below apply to each of the independent Claims 1, 20, 39 and 50.)

INDEPENDENT CLAIM 1

1. ***Fernandez* does not teach "detecting that a portion of a query execution plan… will cause a first producer execution unit, that will perform said portion, to generate XML data for use by a second consumer execution unit in performing another portion of said query execution plan," recited in Claim 1.**

The Office Action alleges that FIGS. 6-7 of *Fernandez* show the following express limitations of Claim 1:

> "detecting that a portion of a query execution plan… will cause a first producer execution unit, that will perform said portion, to generate XML data for use by a second consumer execution unit in performing another portion of said query execution plan,"

This is incorrect. These limitations clearly require:

- a "producer execution unit" that performs one portion of an execution plan
- a "consumer execution unit" that performs a different portion of the same execution plan
- executing its portion of the plan causes the producer execution unit to generate XML data
- the XML data generated by the producer execution unit is used by the consumer execution unit when the consumer execution unit is performing its portion of the execution plan

In contrast, in FIG. 6, *Fernandez* depicts that a query planner partitions an RXL query into sub-trees, a translator decomposes the sub-trees into SQL queries and submits the SQL queries for execution. An RDBMS executes the SQL queries and returns results that are

merged and integrated into an output XML document. (*Fernandez*: Col. 32, ll. 56-57; Col. 38, ll. 15-20)

Fernandez' FIG. 6 differs in many fundamental ways from the limitation against which it is cited. Specifically, FIG. 6 does not have one execution unit involved in performing a query feeding results to another execution unit involved in performing the same query. Further, while FIG. 6 does mention XML, the mentioned XML is not passed from one execution to another execution unit during performance of an execution plan. Rather, the XML mentioned in FIG. 6 is merely the thing into which the query results are ultimately merged or integrated.

Turning now to FIG. 7 of *Fernandez,* it depicts data structures containing information about nested relations that are derived from an RXL query. (*Fernandez*, Col. 40, ll. 24-26) As the query planner partitions an RXL query into sub-tree and the translator decomposes the sub-tree into SQL queries, the information about the nested relations between the variables in the RXL query needs to be preserved so the results returned by the SQL queries can be merged and integrated according to those relations. (*Fernandez*: Col. 40, ll. 40-42) The relations represent a mapping between the deeply nested variables in the RXL query and the variables in the SQL queries. However, the relations are used (by the XML document generator) when all SQL queries finish generating and providing their results, not "when the consumer execution unit is performing its portion of the execution plan," as recited in Claim 1.

The relations themselves are not "for use by a second consumer execution unit in performing another portion of said query execution plan," as recited in Claim 1. In fact, the relations have nothing to do with "XML data, generated by the first execution unit, to be used by a second execution unit in performing another portion of the execution query," as claimed.

a. ***Fernandez'* query execution plan does not have two execution units, executing portions of a query execution plan, where the second unit uses the XML data generated by the first unit, as recited in Claim 1.**

The Office Action alleges that FIGS. 6-7 of *Fernandez* show "two execution units, where the second unit uses the XML data generated by the first unit," as recited in Claim 1. This is incorrect.

FIG. 6 of *Fernandez* depicts that the partitioned SQL queries (i.e. SQL1, SQL2 … SQLk) are sent to one RDBMS for execution. However, FIG. 6 does not show that the SQL queries are executed by two execution units, as recited in Claim 1. Further, FIG. 6 does not show one execution unit that performs one part of the execution plan and another execution that performs another part of the execution plan, as recited in Claim 1. Therefore, FIG.6 does not support the Office allegation that the query execution plan is executed by two execution units, as recited in Claim 1.

FIG. 7 of *Fernandez* depicts the relations that are used to merge results returned when the execution of the execution plan is completed. Indeed, the relations are used when the execution of the query execution plan is completed, not when a part of the execution plan needs to be performed.

Fernandez also describes a query planner, a query translator and a results integrator. However, none of them executes portions of a query execution plan, as recited in Claim 1. *Fernandez'* query planner partitions an RXL query into sub-trees, but does not execute the query execution plan, as recited in Claim 1. The query translator decomposes the sub-trees into SQL queries and submits the SQL queries for execution, but does not execute the queries. The results integrator merges the results returned by the SQL queries into an output XML document, but does not execute the query execution plan, as recited in Claim 1.

Therefore, *Fernandez* does not show a "first producer execution unit… and… a second consumer execution unit," wherein the second consumer execution unit "uses XML data, generated by the first producer execution unit, in performing another portion of said query execution plan," recited in Claim 1.

b. **Execution of the *Fernandez'* query execution plan returns relational data tuples, not XML data, as recited in Claim 1.**

Fernandez' RDBMS receives SQL queries and generates data tuples for the SQL queries by querying the database. *Fernandez'* RDBMS does **not** output or use any "XML data … in performing another portion of said query execution plan," recited in Claim 1. None of the portions of the *Fernandez'* query execution plan generates XML data or uses XML data generated while another portion of the same query execution plan is performed, as recited in Claim 1.

The data tuples generated by the RDBMS are merged and integrated into an output XML document. The output XML document is presented to the user, but is not used by the "second consumer execution unit to execute another portion of the query execution plan," as recited in Claim 1. In fact, the output XML document generated by *Fernandez'* integrator is never used in execution of any portion of the query execution plan, as recited in Claim 1. The output XML document is generated by *Fernandez'* integrator, not an execution unit, as recited in Claim 1, is not used by the "second consumer execution unit to perform another potion of said query execution plan," as recited in Claim 1.

c. **In *Fernandez*, the mapping between the variables in the RXL query and the relational data is not used in "performing another portion of said query execution plan," recited in Claim 1.**

Fernandez' mapping between the deeply nested variables in the RXL query and the variables in the SQL queries is used, **after results of the SQL queries are computed**, to integrate the relational data into the resulting output XML document. Since the results of the SQL query only exist **after** the SQL query has been executed, and the mapping is used **after** the results are computed, then the mapping necessarily is used **after** the SQL queries are computed. Using the mapping after the queries are computed cannot possibly be the same as using data to perform "another portion of said query execution plan," as recited in Claim 1.

In *Fernandez,* there is no XML data "for use by a second consumer execution unit in performing another portion of said query execution plan," recited in Claim 1.

This error has resulted in faulty rejections.

2. ***Fernandez* does not show or suggest "annotating said information with an annotation that causes XML data generated by said first execution unit to be transformed to a canonical form for use by said second execution unit in performing said another portion of said query execution plan," recited in Claim 1.**

The Office Action alleges that *Fernandez* shows the "annotating …" limitation in Col. 6, ll. 61-67; Col. 7, ll. 1-19; and Col. 28, ll. 1-10. (Final Office Action, page 4) This is incorrect. In the columns that are alleged to show this limitation, *Fernandez* describes a View Tree, which does not contain any component that can be used "by said second execution unit in performing said another portion of said query execution plan," recited in Claim 1. Neither *Fernandez'* View Tree nor any component of the View Tree "causes XML data generated by said first execution unit to be transformed to a… form for use by said second execution unit in performing said another portion of said query execution plan," as claimed.

This error has resulted in faulty rejections.

3. ***Fernandez'* annotation does not "cause removal of one or more references to execution unit-specific data that is accessible by the first execution unit but that is not accessible by the second execution unit," as claimed.**

Fernandez' annotations pertain to annotating the View Tree's nodes (*Fernandez*: Col. 36, ll. 20-28), which are never sent to the execution units, and thus, *Fernandez'* annotating cannot "cause removal of one or more references to execution unit-specific data that is accessible by the first execution unit but that is not accessible by the second execution unit," as claimed. *Fernandez'* annotations have no influence on how and in what form the data is exchanged during the execution of the query execution plan, as recited in Claim 1.

This error has resulted in faulty rejections.

Therefore, the Office Action erred in alleging that Claim 1 is anticipated by *Fernandez*. This error has resulted in faulty rejections.

Reconsideration and withdrawal of the rejection is respectfully requested.

INDEPENDENT CLAIMS 39, 45 AND 50

Each of Claims 39, 45 and 50 recites features similar to those in Claim 1. Therefore, for the reasons discussed in Claim 1, reconsideration and withdrawal of the rejections of Claims 39, 45 and 50 is respectfully requested.

DEPENDENT CLAIMS

The pending claims not discussed so far are dependent claims that depend on independent claims that were discussed above. Because each of the dependant claims includes the limitations of claims upon which they depend, the dependant claims are patentable for at least those reasons the claims upon which the dependant claims depend are patentable. Removal of the rejections with respect to the dependant claims and allowance of the dependant claims is respectfully requested. In addition, the dependant claims introduce additional limitations that independently render them patentable.

DEPENDENT CLAIM 2

1. ***Fernandez* does not teach "wherein the step of generating information includes generating information that, prior to annotating said information, would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit," recited in Claim 2.**

The Office Action alleges that *Fernandez* describes the above limitation of Claim 2 in column 6, ll. 61-67, and column 7, ll. 1-19. This is incorrect.

In column 6 (61-67) and column 7 (1-19), *Fernandez* describes that a user application formulates a user XML-QL query, which is sent along with the RXL view to a query composer. The query composer produces a new view query (i.e. RXL query), which is called an executable query. The executable query is sent to a translator that partitions the executable query into a data extraction part (i.e. one or more SQL queries) and an XML construction part (i.e. an XML template). The template is used to merge the results provided by the RDBMS upon finishing the execution of each of the SQL queries and to generate the output XML document.

However, *Fernandez* does not describe two execution units, recited in Claim 2. Further, *Fernandez* does not describe any type of information that would influence the first execution unit as to how to generate XML output so it can or cannot be used by the second execution unit, as recited in Claim 2. None of the XML-QL query, RXL view query, SQL queries and XML template contains "information that, prior to annotating said information, would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit," as recited in Claim 2.

Further, in *Fernandez,* the output XML document is provided to the user, not to second execution unit, as recited in Claim2. The XML document in *Fernandez* is a merged document comprising all the results to be provided to the user, but does not "cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit," as recited in Claim 2.

The Appellants respectfully note that 35 U.S.C. § 132 requires in connection with a rejection that the Director "notify the applicant thereof, stating the reasons for such rejection." Notably, this section is violated if the rejection "is so uninformative that it prevents the applicant from recognizing and seeking to counter the grounds for rejection." (*Chester v. Miller,* 15 USPQ2d 1333 (Fed. Cir. 1990)). This policy is captured in the Manual of Patent Examining Procedure (MPEP). For example, MPEP § 706 states that

"[t]he goal of examination is to clearly articulate any rejection early in the prosecution process so that applicant has the opportunity to provide evidence of patentability and otherwise respond completely at the earliest opportunity." Furthermore, MPEP § 706.020 indicates that: "[i]t is important for an Examiner to properly communicate the basis for a rejection so that the issues can be identified early and the applicant can be given fair opportunity to reply."

The Appellants respectfully submit that for at least these reasons the rejection of Claims 1-50 should be reversed.

VIII. CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted that the rejections of Claims 1-50 lack the requisite factual and legal bases. Appellants respectfully request that the Honorable Board reverse the rejections of Claims 1-50.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Dated: October 6, 2009

/MalgorzataAKulczycka#50496/___
Malgorzata A. Kulczycka
Reg. No. 50,496

2055 Gateway Place, Suite 550
San Jose, California 95110-1089
Telephone: (408) 414-1228
Facsimile: (408) 414-1076

CLAIMS APPENDIX

1. A method comprising the computer-implemented steps of:

detecting that a portion of a query execution plan to service a request for data will cause a first producer execution unit that will perform said portion, according to said query execution plan, to generate XML data for use by a second consumer execution unit in performing another portion of said query execution plan;

generating information to send to said first execution unit to cause said first execution unit to perform said portion of said query execution plan;

wherein said information would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit; and

annotating said information with an annotation that causes XML data generated by said first execution unit to be transformed to a canonical form for use by said second execution unit in performing said another portion of said query execution plan,

wherein said annotating causes removal of one or more references to execution unit-specific data that is accessible by the first execution unit but that is not accessible by the second execution unit.

2. The method of Claim 1, wherein the step of generating information includes generating information that, prior to annotating said information, would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit, and wherein said canonical form is different from said first form.

3. The method of Claim 2, wherein said first form includes information to locate data that is stored in memory that is exclusive to said first execution unit, and wherein said information to locate data stored in said memory cannot be used by said second execution

unit.

4. The method of Claim 1, wherein said request for data is a database query and said plan is a query plan.

5. The method of Claim 4, wherein said information is one or more database commands.

6. The method of Claim 1, wherein said annotation specifies a transformation operator.

7. The method of Claim 6, further comprising the computer-implemented steps of:

executing said transformation operator, by said first execution unit, to transform XML data generated by said first execution unit to said canonical form; and

sending XML data that is transformed by said first execution unit to said second execution unit in said canonical form.

8. The method of Claim 6, wherein said annotation specifies arguments for said transformation operator, to specify said canonical form.

9. The method of Claim 1, further comprising the computer-implemented steps of:

transforming, by said first execution unit, said XML data to said canonical form based on said annotation.

10. The method of Claim 1, wherein the step of annotating includes annotating said information with an operator to transform said XML data to a canonical form in which said XML data is serialized to represent particular data for a particular XML construct and is included in a serialized image that is sent to said second execution unit.

11. The method of Claim 1, wherein the step of annotating includes annotating said information with an operator to transform said XML data to a canonical form which includes an identifier of memory space where data is persistently stored, and wherein said data in said

memory space is accessible by said second execution unit.

12. The method of Claim 1, wherein the step of annotating includes annotating said information with an operator to transform said XML data to a canonical form in which said XML data is compressed according to a particular compression form that said second execution unit is able to decompress.

13. The method of Claim 1, wherein said first execution unit and said second execution unit are different execution units that are executing, in parallel, work associated with servicing said request.

14. The method of Claim 1, wherein said first execution unit and said second execution unit are different execution units that are each executing, on different servers of a distributed database system, work associated with servicing said request.

15. The method of Claim 1, wherein the steps of detecting, generating and annotating are performed by a means that distributes work associated with servicing said request to said first execution unit and said second execution unit, and wherein said first execution unit and said second execution unit are different execution units that are each executing work associated with servicing said request.

16. The method of Claim 15, wherein said first execution unit and said second execution unit are each executing, on different data sources, work associated with servicing said request.

17. The method of Claim 15, wherein said means that distributes work comprises an application server.

18. The method of Claim 15, wherein said means that distributes work comprises an application that manages workload among multiple means for executing said work.

19. The method of Claim 1, further comprising the computer-implemented steps of:

determining said canonical form from information that describes preferences of each of multiple execution units that performs work associated with servicing said request.

20. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 1.

21. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 2.

22. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 3.

23. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 4.

24. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 5.

25. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 6.

26. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to

perform the method recited in Claim 7.

27. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 8.

28. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 9.

29. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 10.

30. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 11.

31. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 12.

32. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 13.

33. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 14.

34. A computer-readable storage medium carrying one or more sequences of instructions

which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 15.

35. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 16.

36. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 17.

37. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 18.

38. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 19.

39. A method for processing XML data, comprising the computer-implemented steps of:

receiving information at a first execution unit to cause said first execution unit to perform work associated with a query execution plan for servicing a request for data;

wherein said information comprises an annotation that causes the XML data generated by said first execution unit to be transformed to a canonical form for use by a second execution unit;

wherein said information, without said annotation, would cause said second execution unit to receive from said first execution unit XML data in a first form that cannot be used by said second execution unit;

transforming XML data generated by said first execution unit to said canonical form prior to providing said XML data to said second execution unit,

wherein transforming XML data comprises removing one or more references to execution unit-specific data that is accessible to the first execution unit but that is not accessible to the second execution unit; and

providing XML data that is transformed to said second execution unit in said canonical form for use in performing work associated with said query execution plan by said second execution unit.

40. The method of Claim 39, wherein the step of transforming said XML data to said canonical form is performed by said first execution unit.

41. The method of Claim 40, wherein the step of transforming comprises executing an operator specified in said annotation.

42. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 39.

43. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 40.

44. A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 41.

45. A database system comprising:

a query optimizer that receives a database query, formulates a query plan based on

said query, and sends information based on said plan to a first execution unit;

wherein formulating a plan includes determining that said first execution unit produces XML data for use by a second execution unit, and determining whether said first execution unit produces said XML data in a first form that said second execution unit is able to use;

said first execution unit that receives said information from said query optimizer; and

said second execution unit that receives said XML data from said first execution unit.

46. The system of Claim 45,

wherein, if it is determined that said second execution unit is able to use said XML data in said first form,

said information that said query optimizer sends to said first execution unit comprises a direction to send said XML data in said first form to said second execution unit;

said first execution unit produces XML data in said first form while servicing said query, and sends said XML data to said second execution unit; and

said second execution unit receives said XML data in said first form, and services said query based on said XML data in said first form.

47. The system of Claim 45,

wherein, if it is determined that said second execution unit is unable to use said XML data in said first form,

said information that said query optimizer sends to said first execution unit comprises transformation information that causes said first execution unit to transform said XML data that is produced by said first execution unit to a second form that said second execution unit

is able to use;

said first execution unit produces transformed XML data in said second form based on said transformation information while servicing said query, and sends said transformed XML data to said second execution unit; and

said second execution unit receives said transformed XML data in said second form, and services said query based on said transformed XML data.

48. The system of Claim 45, wherein said first execution unit and said second execution unit are different execution units that are servicing said request by performing work in parallel.

49. The system of Claim 45, wherein said first execution unit and said second execution unit are different execution units that are servicing said request by performing work on different servers of a distributed database system.

50. A system comprising:

means for detecting that a portion of a query execution plan to service a request for data will cause a first producer execution unit that will perform said portion, according to said query execution plan, to generate XML data for use by a second consumer execution unit in performing another portion of said query execution plan;

means for generating information to send to said first execution unit to cause said first execution unit to perform said portion of said query execution plan;

wherein said information would cause said first execution unit to generate said XML data in a first form that cannot be used by said second execution unit; and

means for annotating said information with an annotation that causes XML data generated by said first execution unit to be transformed to a canonical form for use by said

second execution unit in performing said another portion of said query execution plan,

wherein said annotating causes removal of one or more references to execution unit-specific data that is accessible by the first execution unit but that is not accessible by the second execution unit.

EVIDENCE APPENDIX

No evidence has been submitted under 37 CFR 1.130, 1.131, or 1.132. No other evidence has been entered by the examiner and relied upon by Appellant in the appeal. Therefore, no copies of any evidence are deemed necessary.

RELATED PROCEEDINGS APPENDIX

No decisions have been rendered by any court or the Board in any proceeding identified in the Related Appeals and Interferences section of the brief. Therefore, no copies of any decisions are deemed to be necessary.